Cost-based attribute selection for GRE (GRAPH-SC/GRAPH-FP)

Mariët Theune **Pascal Touset**
Human Media Interaction
University of Twente
Enschede, The Netherlands
m.theune@utwente.nl
p.b.touset@student.utwente.nl

Jette Viethen
Centre for Language Technology
Macquarie University
Sydney, Australia
jviethen@ics.mq.edu.au

Emiel Krahmer
Communication & Cognition
University of Tilburg
Tilburg, The Netherlands
e.j.krahmer@uvt.nl

1 The Challenge

In this paper we discuss several approaches to the problem of content determination for the generation of referring expressions (GRE) using the Graph-based framework of Krahmer et al. (2003). This work was carried out in the context of the *First NLG Shared Task and Evaluation Challenge on Attribute Selection for Referring Expression Generation*.

In the *shared task proper* of the Challenge the output of submitted systems is evaluated against a corpus of human-produced descriptions for the same input. The data used for training, development and testing is a subset of the Aberdeen TUNA Corpus, a human-generated data set designed for the attribute selection task. The data set is divided into a Furniture domain and a People domain, focussing on descriptions of singular entities. Attributes in both domains are type, orientation and x- and y-dimensions. Additional attributes in the Furniture domain are colour and size, and additional attributes in the People domain are age, orientation, hair colour, and the presence or absence of hair, beards, glasses and different clothing items. Each input is a pair consisting of the XML representation of the attributes of all objects contained in a scene and a pointer indicating the intended referent. The output descriptions of the target are lists of attributes, also encoded in XML.

Evaluation of the submitted systems is based on the Dice coefficient of similarity between system output and the human-produced data from the TUNA Corpus. There will also be a small-scale task-based evaluation with human participants.[1]

[1]For details on the TUNA corpus and the GRE Challenge see http://www.csd.abdn.ac.uk/research/tuna/corpus/ and http://www.csd.abdn.ac.uk/research/evaluation/index.php

2 The Graph-based Algorithm

The graph-based GRE algorithm represents the domain of conversation as a labelled directed graph, in which the domain objects are modelled as the vertices, properties are modelled as looping edges, and relations between objects are modelled as edges between the corresponding vertices. To generate a distinguishing description, the graph-based algorithm searches for a subgraph of the domain graph that uniquely refers to the target object.

Informally, a subgraph refers to the target object if and only if it can be 'placed over' the domain graph in such a way that the subgraph vertex representing the target object can be 'placed over' the vertex of the target in the domain graph, and each of the labeled edges in the subgraph can be 'placed over' a corresponding edge in the domain graph with the same label. Furthermore, a subgraph is distinguishing if and only if it refers to exactly one vertex in the domain graph. The informal notion of one graph being 'placed over' another corresponds with the well-known mathematical graph construction called subgraph isomorphism.

Generally there is more than one distinguishing graph referring to an object. To give preference to some solutions over others, the algorithm uses cost functions that assign costs to the edges of a graph. The total cost of a graph is determined by summing over the costs associated with its edges, and the algorithm always returns the cheapest graph it can find. There are numerous ways in which costs can be as-

signed to edges; below we discuss the cost functions we investigated for the GRE Challenge.

3 The Cost Functions

Using the GRE Challenge training set, we experimented with various cost functions: some that were suggested in Krahmer et al. (2003), and some that we specifically created for the Challenge. Since people tend to mention the type of an object regardless whether it is a distinguishing property, we modified the graph-based algorithm so that type is selected by default at zero cost for all cost functions.

Simple Costs: In our simplest cost function, all attributes except type cost 1. Using Simple Costs the graph-based algorithm resembles the Full Brevity algorithm of Dale (1992) in that a shortest distinguishing description is always favoured.

Absolute versus Relative: Here, absolute properties are cheaper (cost 1) than relative ones (cost 2), mimicking the human preference for absolute properties over relative ones (Dale and Reiter, 1995). In the Furniture domain the relative properties are size and dimension; in the People domain only dimension is relative. All other properties are absolute.

Stochastic Costs: In this function, the more frequently a property occurs in the corpus of human descriptions, the cheaper it is. Property costs in this function have been defined as $Cost(e) = -\log_2(P(e))$ (Krahmer et al., 2003), where $P(e)$ is the probability that property e (corresponding to an edge in the scene graph) occurs in a description, given that the target object actually has this property. The probability $P(e)$ is estimated by determining the frequency of each property in the description corpus, relative to the number of target objects with this property.

Expensive Dimensions: Based on the informal observation that dimensions seem to be used less frequently than other attributes, we made the x- and y-dimension attributes cost 2, and all other attributes cost 1. In the People domain, this cost function is the same as the Absolute versus Relative function, since in that domain the x- and y-dimensions are the only relative attributes.

Free Properties: For each domain we designed a domain-specific cost function simulating the human tendency to add redundant properties to object descriptions. In the Furniture domain, most descriptions mention colour even when this is not needed to identify the target object. Therefore, the Free Properties function for this domain allows the colour property to be added to a description at zero cost. In the People domain, beards and glasses tend to be mentioned even when strictly spoken they are redundant. So, in the Free Properties function for this domain, the properties hasBeard=1 and hasGlasses=1 can be added to a description at zero cost.

We made two versions of Free Properties: in Free Properties 1 all non-free properties have the same costs as in the Stochastic cost function, and in Free Properties 2 all non-free properties have the same costs as in Expensive Dimensions.

	Furniture		*People*	
Cost function	**Mean**	PRP	**Mean**	PRP
Simple Costs	0.53	5.0	0.59	16.5
Abs. vs Rel.	0.59	13.0	0.61	21.8
Exp. Dimensions	0.67	16.7	0.61	21.8
Stochastic Costs	0.69	25.1	0.65	22.8
Free Properties 1	0.75	31.4	0.67	24.8
Free Properties 2	0.77	39.8	0.63	18.9

Table 1: Results for the training sets

4 Results

Table 1 shows the results of the different cost functions on the Furniture and People training sets. We provide both mean Dice score and Perfect Recall Percentage (PRP), which is the proportion of Dice scores of 1 (Gatt et al., 2007). For both domains, the two Free Properties functions score best, and the Simple Cost function scores worst.

Table 2 shows the scores on the development sets. Again, the two Free Properties functions score best in both domains. Interestingly, the Stochastic cost functions, which were based on the statistics from the training corpus, turn out to do very well on the development sets too. Thus, there appears to be no overfitting on the training corpus. Remarkable is also that in the People domain, the Simple Costs function has exactly the same scores as the more sophisticated Absolute versus Relative and Expensive Dimensions functions (which are identical), even though it does not always generate the same descriptions.

	Furniture		*People*	
Cost function	**Mean**	PRP	**Mean**	PRP
Simple Costs	0.55	3.8	0.63	22.1
Abs. vs Rel.	0.64	16.3	0.63	22.1
Exp. Dimensions	0.67	21.3	0.63	22.1
Stochastic Costs	0.66	20.0	0.66	22.1
Free Properties 1	0.71	30.0	0.67	25.0
Free Properties 2	0.72	31.3	0.65	25.0

Table 2: Results for the development sets

In our submission to the GRE Challenge we will use the following cost functions: Stochastic Costs (GRAPH-SC) and Free Properties 1 (GRAPH-FP), as these performed best on the training and development sets for both domains.

5 Discussion

Best-scoring on the training and development sets are the Free Properties functions, which mimic people's tendency to add redundant properties to descriptions. This tendency formed the basis for the Incremental Algorithm of Dale and Reiter (1995) and was also observed by Viethen and Dale (2006) in human-produced descriptions of coloured drawers. One possible explanation why people add redundant information is that they do it to enhance clarity of the description, at the cost of brevity (Khan et al., 2006). Apparently, our Free Properties cost functions approximate the balance between clarity and brevity found in human descriptions.

To get an impression of how well the graph-based algorithm performs compared to other GRE algorithms, we compare our results to those of Gatt et al. (2007). They evaluated some classic GRE algorithms using a data set from the TUNA corpus (Furniture domain) that included all attributes used in the GRE Challenge. The best scoring algorithm on this data set was the Incremental Algorithm with the following attribute ordering: colour, y-dimension, orientation, size, x-dimension. The mean Dice score for this algorithm was 0.64, which is only slightly worse than most of our scores for the Furniture domain. However, the PRP was only 10%, which is much lower than our PRP scores. Possibly, this difference has to do with the different description strategies people use. For example, we found that 68.8% of the descriptions in the People domain do not include dimension properties at all, whereas 23.9% consist *only* of dimensions (besides type). Because dimensions are relatively expensive in all our cost functions, our algorithm may be more tuned to the first strategy than the latter. The algorithms used by Gatt et al. (2007) appear to be more neutral in this respect. Given that the choice of description strategy mostly seems to be a matter of personal preference, it will be hard for our algorithm to always select the 'correct' strategy (i.e., the one picked by the human subject). This problem might be solved by using speaker-dependent cost functions, which could be derived from the TUNA corpus. However, this will not work when trying to simulate descriptions produced by 'new' speakers.

Another way to achieve better scores would be to use cost functions that capture dependencies between attributes. From informal observations of the corpus it seems that certain properties are often mentioned in combination, e.g., hairColour and hasHair in the People domain. More detailed statistical analysis of the corpus data is required to identify the most promising combinations for each domain.

References

Robert Dale. 1992. *Generating referring expressions: constructing descriptions in a domain of objects and processes.* The MIT Press, Cambridge, Massachusetts.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263.

Albert Gatt, Ielka van der Sluis and Kees van Deemter 2007. Evaluating algorithms for the generation of referring expressions using a balanced corpus. *Proceedings of ENLG-2007*, 49–56.

Imtiaz Hussain Khan, Graeme Ritchie and Kees van Deemter. 2006. The clarity-brevity trade-off in generating referring expressions. *Proceedings of INLG-2006*, 84–86.

Emiel Kramer, Sebastiaan van Erk and André Verleg. 2003. Graph-based generation of referring expressions. *Computational Linguistics*, 29(1):53–72.

Jette Viethen and Robert Dale. 2006. Algorithms for generating referring expressions: Do they do what people do? *Proceedings of INLG-2006*, 63–70.